Automated Planning for Healthcare Logistics

Academic Year 2024-2025
University of Trento - Artificial Intelligence Systems

Chiara Musso
chiara.musso@studenti.unitn.it

Emmanuele V. Coppola
emmanuele.coppola@studenti.unitn.it

March 8, 2025

Abstract

This report presents the implementation of an automated planning solution for a healthcare logistics scenario using PDDL/HDDL and the ROS2-based PlanSys2 framework. The project involved modeling and solving sequential planning problems of increasing complexity, incorporating hierarchical task networks, durative actions, and multi-agent coordination. Solutions were tested using state-of-the-art planners such as Fast Downward and Optic. The results demonstrate the feasibility of automated planning in healthcare logistics, highlighting key challenges and opportunities for optimization.

1. Introduction

This report explores the application of automated planning in a healthcare logistics scenario. Automated planning is a branch of Artificial Intelligence that determines action sequences to achieve specific goals. In this case, we model a healthcare facility where robotic agents transport medical supplies and accompany patients to designated units.

The problem involves a structured environment where medical units require essential supplies, and patients must be escorted to appropriate locations. Robotic agents operate under spatial and capacity constraints, executing tasks such as filling, loading, moving, and delivering supplies. Planning is performed offline, ensuring efficient task execution without real-time computational limitations.

The assignment consists of five progressively complex problems:

- **Problem 1**: Basic logistics planning for medical supply delivery and patient transport.
- **Problem 2**: Introduction of carriers with load capacity constraints.
- **Problem 3**: Implementation of **Hierarchical Task Networks (HTN)** using HDDL.
- **Problem 4**: Incorporation of **durative actions** for temporal constraints and concurrency.
- **Problem 5**: Integration with **PlanSys2** for execution in a simulated robotic environment.

This report follows the structure of the assignment. Section 2 introduces PDDL and HDDL, Section 3 describes the problem and design choices, Section 4 presents results, Section 5 details the project structure, and Section 6 concludes with final considerations. All related code is available in the GitHub repository.

2. PDDL and HDDL

Planning Domain Definition Language (PDDL) is a standardized language for defining planning tasks and domains in artificial intelligence. It enables automated planners to generate action sequences based on domain constraints and goal conditions. This assignment required modeling a logistics scenario using PDDL, progressively introducing more advanced planning techniques.

A PDDL model consists of:

- **Domain file**: Defines object types, predicates, and actions that structure the planning environment.
- **Problem file**: Specifies the initial state and goal conditions for a specific planning instance.

2.1. Hierarchical Task Networks (HTN)

In **Problem 3**, the planning model was extended using **Hierarchical Task Networks (HTN)** through **HDDL**. Unlike classical PDDL, which defines flat action sequences, HDDL enables **task decomposition**, where high-level tasks break down into structured subtasks. This method allows for more efficient multi-agent coordination and structured execution of complex objectives.

2.2. Planning System Requirements

The PDDL models developed in this assignment incorporated the following key requirements:

- **STRIPS**: Ensuring fundamental preconditions and effects for action execution.

- **Typing**: Defining structured object categories (e.g., robots, boxes, patients, locations).
- **Durative Actions**: Implementing time-dependent constraints to allow concurrent and sequential execution (Problem 4).

The final implementation (Problem 5) was deployed in **PlanSys2**, demonstrating the integration of PDDL-based planning in robotic systems. The following sections provide a detailed analysis of the domain and problem files developed for each scenario.

3. Problems and Design Choices

This section outlines the exercises conducted in this project, focusing on automated planning for real-world scenarios involving robotic agents. The main challenges include handling logistics constraints, managing task execution, and ensuring efficient resource utilization. **PDDL and HDDL** were used to formally define and solve these problems.

3.1. Scenario and Assumptions

The planning problem is inspired by a healthcare logistics scenario where robotic agents must transport medical supplies and accompany patients within a structured healthcare facility. The facility consists of multiple medical units, each dedicated to specific treatments, requiring essential medical devices and drugs for their operations. Additionally, patients must be escorted to their assigned medical units.

To ensure a structured and scalable planning model, we define the following assumptions:

- **Medical Units and Locations**: Each medical unit is at a fixed location, but multiple units can exist in the same location.
- **Boxes and Medical Supplies**:
  - Each box is initially stored at a specific location (e.g., the central warehouse).
  - Boxes contain medical supplies such as scalpels, tongue depressors, or aspirin.
  - Boxes can be emptied at a medical unit, transferring the contained items.
  - The model allows managing types of medical supplies to be problem dependent and not domain dependent.
- **Tracking of Supplies**:
  - The system must track whether each medical unit has it's order of medical supplies satisfied.
  - Since multiple medical units can share the same location, tracking must be content-based rather than location-based.
- **Robotic Agents Capabilities**:
  - Robots can unload a box, delivering its contents to the current location's medical unit.
  - Depending on the assignment robots can pick up and transport a single box at a time.
  - Robots can move between predefined connected locations within the facility.
  - Robots can deliver a box to a medical unit at the same location.
- **Navigation Constraints**:
  - Robots can only move between connected locations as per facility layout. The robot type defines which connection an agent can use.
  - The movement structure mimics a real hospital, where paths and access points define viable routes.
- **Patient Transportation**:
  - Patients start at the entrance of the healthcare facility.
  - Only specific accompanying robots can transport patients.
  - Each accompanying robot can escort only one patient at a time.
  - Robots for patient transport are distinct from those handling medical supplies.

The following subsections describe the different problem formulations, progressively increasing in complexity from basic logistics planning to hierarchical and temporal planning approaches.

3.2. Problem 1: Healthcare Logistics

The first problem involves a **healthcare logistics scenario**, where robotic agents transport medical supplies and escort patients within a structured environment. The objective is to ensure medical units receive required supplies while patients reach their assigned locations.

3.2.1 Problem Definition

The scenario consists of:

- A **central warehouse**, where all supplies are initially stored.
- Multiple **medical units**, each requiring specific types of supplies.
- An **entrance**, where patients await transportation.
- Two types of robots: a **delivery robot** for supply transport and an **accompanying robot** for patient escort.

At the start, all supply boxes are in the central warehouse, and robots must efficiently coordinate their actions to deliver supplies and transport patients.

3.2.2 Assumptions

- There is no distinctions between paths that can take accompanying robot and paths that only delivery robots can take.
- A standard delivery robot can deliver only one box at a time.
- There is no distinction between patient type (we don't consider wheelchair needing patients).
- A patient will always be with the same robot through the whole path to the medical unit.
- There is no need to track the position of a single object only the box that contains it.

3.2.3 Modeling Approach

A **PDDL domain** was designed with:

- **Types**: locations, supply boxes, medical supplies, patients, and robots.
- **Predicates**: tracking positions of robots, boxes, and patients.
- **Actions**: defining movement, loading/unloading, and patient transport.

The problem instance includes **three medical units**, **two patients**, and **five supply boxes**. The planning challenge is to efficiently allocate robots to tasks while ensuring correct supply types are delivered.

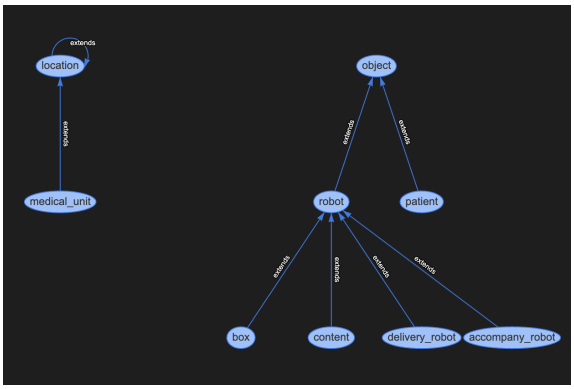

Figure 1: Domain hierarchy problem 1.

3.3. Problem 2: Carriers and Capacity Constraints

This problem extends **Problem 1** by introducing **carriers** for medical supply transport. Unlike before, where a robot could carry only one box at a time, robots now have carriers with limited capacity.

3.3.1 Key Modifications

- Each **robot** has a **carrier** with a fixed capacity.
- Robots can load multiple boxes up to their capacity.
- Supply deliveries must be planned to optimize transport routes.
- Both **terrestrial robots** and **drones** participate in supply transport.

3.3.2 Modeling Approach

The **PDDL model** was extended to include:

- **Carrier tracking**: Ensuring that a carrier will not transport more cargo than it can.
- **Fluents**: monitoring the number of boxes loaded.
- **New predicates**: specifying loaded boxes and their locations.
- **Optimized actions**: enabling robots to load, transport, and unload multiple boxes.

To model a realistic problem, we used data from the "Ospedale dei bambini Vittore Buzzi". We found online the planimetry of the hospital and used the online application geojson.io to calculate some basic distances between sectors and key points.

The problem instance includes **four carriers**, **two delivery robots**, **two drones**, **ten supply boxes** and **one accompanying robot**. Robots can transport multiple boxes before returning to the warehouse, introducing a **logistics optimization challenge**.

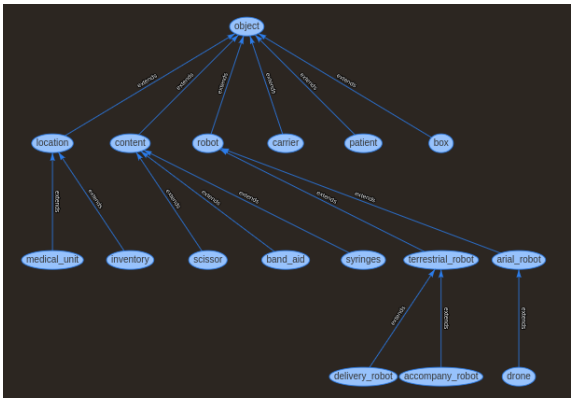

Figure 2: Domain hierarchy problem 2.

3.4. Problem 3: Hierarchical Task Networks (HTN)

This problem extends **Problem 2** by introducing **Hierarchical Task Networks (HTN)**, a structured planning approach that decomposes high-level objectives into a hierarchy of tasks and subtasks. This hierarchical decomposition enables a modular and scalable representation of the healthcare logistics problem.

3.4.1 Problem Definition

The scenario remains consistent with **Problem 2**, where robotic agents deliver medical supplies and escort patients within a healthcare facility. However, instead of using a flat PDDL action model, we utilize **Hierarchical Domain Definition Language (HDDL)** to represent the problem in terms of hierarchical tasks.

The key components of the HTN model are:

- **High-level tasks**: Represent the main objectives, such as delivering medical supplies or escorting patients.
- **Methods**: Specify how each high-level task decomposes into ordered sequences of subtasks.
- **Task decomposition**: Ensures structured execution by breaking complex actions into more granular operations.

3.4.2 Assumptions

The HTN formulation adheres to the same fundamental assumptions as **Problem 2**, with additional constraints imposed by the hierarchical structure. The key assumptions include:

- The same robotic agents, carriers, and supplies as defined in **Problem 2** are used.
- Each high-level task must be decomposed into a sequence of primitive actions.
- **Movement tasks** are handled differently for terrestrial and aerial robots.
- **Supply transportation and patient escorting** follow structured execution sequences.

3.4.3 HTN Task Representation

The hierarchical model introduces the following **HTN tasks**, each corresponding to an essential operation in the logistics system:

- **get-to (?r - robot ?to - location)**: Moves a robot to a specified location.
- **deliver-patient (?r - accompany_robot ?p - patient ?m - medical_unit)**: Escorts a patient to their assigned medical unit.
- **move_patient (?r - accompany_robot ?p - patient ?l1 ?l2 - location)**: Handles patient movement between intermediate locations.
- **load_any_box_to_carrier (?r - robot ?c - carrier ?l - inventory ?b - box)**: Loads a box into a carrier at an inventory location.
- **unload_box_to_location (?r - robot ?c - carrier ?b - box ?l - location)**: Unloads a box at the designated destination.

The hierarchical decomposition is performed using **HTN methods**, which structure how tasks are broken down:

- **m-move-flying**: Moves an aerial robot between locations via a drone port.
- **m-move-terrain**: Moves a terrestrial robot between physically connected locations.
- **m-deliver-patient**: Decomposes the patient delivery process into sequential escorting actions.
- **m-load-any-to-carrier**: Selects and loads a supply box into an appropriate carrier.
- **m-unload-box-to-location**: Unloads a transported supply box at a medical unit or designated inventory.
- **m-deliver-box**: Integrates loading, movement, and unloading operations into a structured process.

3.4.4 HTN Problem Instance

The HTN problem instance retains the core elements of **Problem 2** while structuring the plan execution hierarchically. The primary workflow consists of:

- Moving robots to their required locations.
- Loading medical supplies into carriers.
- Transporting supplies to designated medical units.
- Unloading supplies at the correct locations.
- Escorting patients from the entrance to their assigned medical units.

3.4.5 Key Challenges and Considerations

The introduction of HTN introduces several new challenges that must be addressed:

- **Increased complexity**: Task decomposition requires careful structuring to avoid redundant or unnecessary operations.
- **Task sequencing constraints**: Ensuring that subtask execution follows logical constraints and respects real-world limitations.
- **Parallel execution**: Balancing concurrent execution of patient transport and supply delivery while ensuring correct task dependencies.

The structured decomposition provided by the HTN approach allows for a more scalable and modular solution to the healthcare logistics problem, providing advantages over traditional flat PDDL formulations.

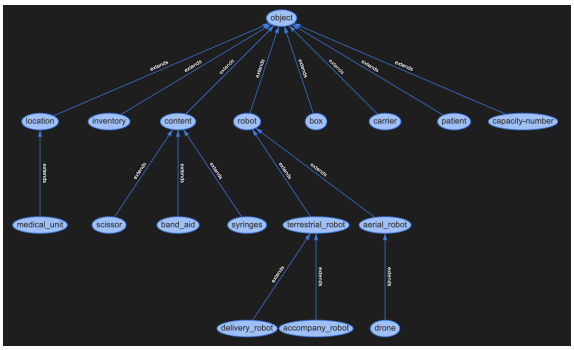

Figure 3: Domain hierarchy for Problem 3.

3.5. Problem 4: Durative Actions and Time Constraints

This problem extends **Problem 2** by introducing **durative actions** and temporal constraints, ensuring that tasks such as movement, loading, and unloading occur over a defined duration. The goal is to optimize the overall execution time while maintaining parallel task execution whenever feasible.

3.5.1 Key Modifications

- **Durative actions**: Each action has a defined execution time.
- **Parallel execution**: Independent tasks can be executed simultaneously.
- **Speed-based movement**: Travel time is determined by robot speed and distance.

3.5.2 Modeling Approach

The **PDDL model** is extended with durative actions, modifying movement, loading, unloading, and patient transport:

- **Timed movement**: Travel time is computed based on speed and distance.
- **Fixed action durations**: Loading, unloading, and patient escorting require specific times.
- **Fluents**: Added functions to track speed, distances, and execution times.

To generate a realistic problem instance to resolve we found data about the children's hospital "Vittore Buzzi". We modeled the connection and the distances used in the problem based on the planimetry found online and OpenStreetMap data using the online application geojson.io as seen in 4.

The problem instance consists of:

- **Two delivery robots**, **two drones**, and **one accompanying robot**.
- **Four carriers** with predefined capacities.
- **Ten supply boxes** containing aspirin, scalpels, and tongue depressors.
- **Predefined movement speeds** and **location distances** for realistic execution times.

3.5.3 Key Considerations

The introduction of temporal constraints impacts the planning process in several ways:

- **Realistic execution times**: Actions now require time to complete.
- **Optimized scheduling**: The planner must minimize overall execution time.
- **Parallel task execution**: Independent tasks can be performed simultaneously.

For example, while a drone can transport a box between two locations, a delivery robot may simultaneously unload medical supplies at a unit. However, physical constraints prevent actions like picking up multiple boxes at once or loading while moving.

By integrating durative actions, the planner must not only generate a feasible plan but also optimize task execution to reduce total mission time.

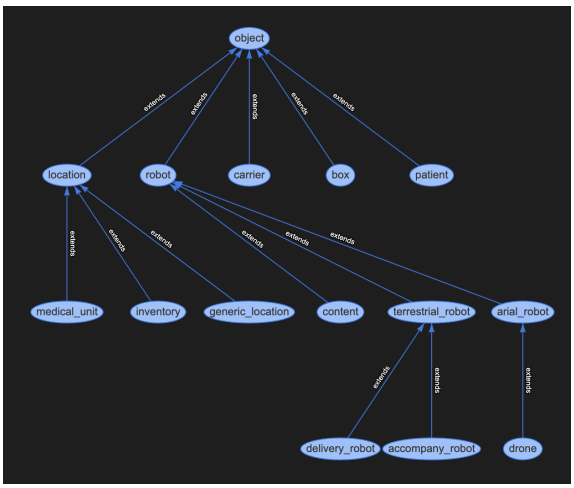

Figure 5: Domain hierarchy problem 4.

3.6. Problem 5: Implementation in PlanSys2 with Fake Actions

This problem focuses on implementing the final planning scenario within **PlanSys2**, based on the outcome of **Problem 4**. The implementation follows the PlanSys2 framework and incorporates **fake actions**, which allow structured representation of planning tasks, enabling simulation of complex behaviors without requiring fully implemented effects for every action. This

Figure 4: On the left, the defined main locations; on the right, the original planimetry.

approach ensures modular execution while maintaining compatibility with the previous problems.

3.6.1 Key Modifications

- **Removal of type hierarchy**: All object types were flattened, requiring explicit predicates for each robot and location type.
- **Elimination of numeric fluents**: Numerical constraints, such as capacity management and duration calculations, were replaced with discrete predicates.
- **Predicate-based capacity tracking**: Robot carrier capacities and medical unit inventory levels were handled using predefined predicates instead of numeric values.
- **Fixed-duration actions**: Since PlanSys2 does not support dynamically computed durations, all actions were assigned fixed time values.
- **Segmentation of movement actions**: Separate movement actions were created for each robot type to ensure domain compatibility.
- **Fake actions for task simulation**: Actions such as patient transport and box handling were restructured to be compatible with PlanSys2 execution.

3.6.2 Modeling Approach

The **PDDL model** was adapted to align with the limitations of PlanSys2 while preserving the logic from Problem 4. The following changes were implemented:

- **Explicit movement predicates**: Since type hierarchies were removed, separate predicates were introduced for tracking delivery robots, accompanying robots, and drones in various locations.
- **Discrete capacity management**: Carrier capacities were managed through predicates such as `carrier_capacity`, ensuring that box loads could be tracked without using numeric fluents.
- **Fake actions for patient transport**: The patient transportation process was restructured into discrete steps that could be simulated within PlanSys2 without modifying its execution framework.
- **Location-specific movement actions**: Each movement action was specialized based on the location type, requiring explicit transitions between inventory locations, medical units, and generic locations.
- **Explicit load/unload operations**: Loading and unloading actions were modified to ensure compatibility with predicate-based inventory tracking.

To ensure full compatibility with Problem 4 files, the domain was adapted to maintain structural consistency. The removal of the type hierarchy required a manual restructuring of movement, storage, and delivery actions to function correctly in PlanSys2.

3.6.3 Key Considerations

The adaptation of the planning model to PlanSys2 introduced several constraints that influenced its structure and execution:

- **Simplified execution model**: Fake actions allowed for structured execution without requiring modifications to the robotic system.

- **Fixed time constraints**: Since PlanSys2 does not support numeric calculations for durations, all action times were manually assigned.
- **Compatibility with PlanSys2**: The planning model was adjusted to ensure it could be executed within the constraints of the POPF 1.1 planner.

By restructuring the planning domain to fit within the constraints of PlanSys2, the implementation maintains logical coherence while ensuring that healthcare logistics tasks can be executed efficiently.

4. Results

In this section, we discuss the resulting plans obtained for each of the defined tasks. These plans are available in the computed plans folder in the GitHub repository. Additionally, a summary of the results is presented at the end of the GitHub read-me file. All the the following test numbers derives from executing code using a M3 equipped MacBook PRO.

4.1. Problem 1

The plan related to Problem 1 has been computed using the **Fast Downward** planner from the Planutils suite.

The computed plan follows a **structured execution pattern**. The first set of actions focuses on **patient transportation**, where the accompanying robot picks up each patient from the entrance and escorts them to their assigned medical unit. The robot then returns to the entrance to repeat the process for the next patient. Once all patients have been transported, the second phase begins, where the **delivery robot** is responsible for distributing medical supplies. It loads supply boxes from the **central warehouse**, moves to the required medical units, and unloads both the boxes and their contents. After each delivery, the robot returns to the warehouse to reload for the next supply run.

This execution structure ensures that all tasks are completed in a straightforward and logical sequence. The final plan was computed efficiently, with a **total execution cost of 37**. The search process terminated quickly as no further improvements could be made, suggesting that the obtained solution is near-optimal. The final statistics indicate that the planner **generated 571 nodes and expanded 361 nodes**, with a total of 132 match tree nodes. The entire search process was completed in **0.001645 seconds**.

While the plan successfully achieves all goals, some **inefficiencies** were observed, particularly in **redundant movements** where the delivery robot returns to the warehouse for each new box instead of optimizing multi-box deliveries. Future refinements could explore strategies to **minimize unnecessary movements** and leverage **carrier-based optimizations**, as introduced in the following problems.

4.2. Problem 2

The plan related to Problem 2 has been computed using the **Fast Downward** planner from the Planutils suite. This problem introduces the use of **carriers**, allowing **delivery robots** to transport **multiple boxes in a single trip**, reducing the number of required movements.

The first computed plan already showed **significant improvements** compared to Problem 1. The sequence of actions begins with the **transportation of patients**, ensuring they reach their respective medical units before logistics operations commence. The delivery robots then proceed with the **loading of medical supply boxes** into carriers, making optimal use of their capacity. The **movement between locations is structured efficiently**, avoiding unnecessary trips back to the warehouse unless absolutely necessary. At each medical unit, the robots unload the correct supplies and, when needed, reload additional boxes for subsequent deliveries.

While the first plan demonstrated a **considerable optimization**, minor inefficiencies in movement paths were observed. A **refined version of the plan** was later computed, adjusting the order of certain deliveries and improving resource utilization. The structured use of carriers allowed for a **notable reduction in redundant movements**, making the process **significantly more efficient** than in the previous problem. The final computed plan is **highly optimized**, achieving the goals with a minimal number of actions and efficient task execution.

The final search statistics show that the plan was computed in **0.003012 seconds**, with a total of **476 match tree nodes, 545 nodes generated, and 388 expanded**. The final cost of the plan was **12**. Although the planner did not explicitly confirm optimality, by analyzing the structure of the plan, it appears that a more efficient solution is unlikely to exist. Future refinements could focus on **fine-tuning load balancing** between the delivery robots and further **optimizing movement paths**.

4.3. Problem 3

The plan for **Problem 3** was computed using the **PANDA** planner, applying **Hierarchical Task Networks (HTN)** to structure the execution of healthcare logistics tasks. This approach ensures a modular decomposition of complex operations, breaking high-level tasks into structured subtasks that optimize robot coordination and movement.

The computed plan follows a well-defined sequence. The execution begins with **patient transportation**, ensuring that all patients are escorted to their assigned medical units before initiating the supply deliveries. Accompanying robots first take charge of patients at the entrance and navigate through the facility until they reach the designated medical units. Once

all patients have arrived, the **delivery robots begin the supply transportation process**, loading medical supply boxes into carriers while considering capacity constraints. Drones and ground robots efficiently navigate between locations, minimizing unnecessary movements and ensuring a smooth task execution.

The final computed plan consists of **243 actions**, of which **187 are primitive actions** and **56 are abstract tasks**. The hierarchical decomposition involved **190 methods**, leading to **30 task decompositions** during execution. The planner explored **11,165,621 search nodes**, completing the computation in **58,525 milliseconds**. The final execution sequence resulted in **28 primitive tasks**, with a structured decomposition that allowed for an efficient and synchronized completion of all objectives.

The search process demonstrated high efficiency, reaching a peak performance of **207,415 nodes per second**, with a total of **10,495,764 generated nodes**. The **fringe size** peaked at **2,097,475**, while the **modification depth varied between 31 and 55**, depending on the complexity of the task.

While the initial plan was well-structured, minor inefficiencies in movement paths were identified. A refined version of the plan was computed, optimizing the order of movements and improving the coordination between aerial and terrestrial robots. This resulted in reduced redundant actions and a better balance between different robot types. The structured nature of the HTN approach allowed for **modular execution**, ensuring that supply deliveries were performed efficiently after patient transport was completed.

The solution sequence computed by PANDA follows a logical progression. First, the accompanying robots take charge of the patients at the entrance and move them to their respective medical units. **accompany_robot2** transports **ciro** to **neuro_surgery**, while **accompany_robot1** escorts **rocco** to **cardiology**. Once the patients are in their designated locations, the delivery phase begins. **drone1** loads **box3** at the **central_warehouse**, moves to **heliport_alpha**, and successfully unloads the supplies. Meanwhile, **delivery_robot1** loads **box1**, moves towards **cardiology**, and unloads the supplies there. **delivery_robot2** follows a similar process, transporting **box2** to **neuro_surgery**.

4.3.1 Complex encoding experiment

We also tried to implement a **more complex encoding of the domain** since we wanted to express the medical unit order as a single task like for example `"deliver #of_aspirins aspirin day_care"` and let the planner determine which robot and which method of delivery to take.

While developing this kind of tasks we encountered some limitation of the PANDA planner in which the planner was capable of only computing some of the tasks and not all of them together. We tried in fact to execute at the same time the tasks of taking patients and the tasks of delivering 2 packages. The result was that PANDA started to calculate Billions of nodes, not reaching solution. If instead we executed the task independently in different executions of PANDA, delivering packages before and than bringing patients at destination, the plan was found both cases near instantaneously. Since these limitations, we preferred to define the simpler encoding as our solution for the assignment and not complete the more complex one and brought us to the following considerations.

4.3.2 Final consideration of hierarchical solutions

Through experimentation, we observed that **PANDA performs tree search but it's not a breadth-first search**. This means that instead of exploring all nodes at a given depth before moving deeper, the planner follows a hierarchical expansion strategy that prioritizes task decomposition and goal achievement. This approach influences the efficiency and structure of the generated plans, allowing for a more direct resolution of high-level tasks but potentially limiting exhaustive search capabilities.

The final execution confirms the advantages of using HTN-based planning over flat PDDL formulations. By structuring the logistics execution hierarchically, the system achieves a modular and scalable solution, ensuring logical task sequencing and minimizing inefficiencies in robotic movement and coordination.

4.4. Problem 4

The plan related to Problem 4 has been computed using the **OPTIC** planner, which supports **durative actions** and **time constraints**. Unlike the previous problems, where actions were treated as instantaneous, this problem introduces **explicit durations for each action**, optimizing execution time through **parallel scheduling**.

The first computed plan efficiently **schedules actions while respecting timing constraints**. The execution begins with **patient transportation**, ensuring that each patient is escorted to their designated medical unit while considering the interaction time required for the escorting process. Simultaneously, the **delivery robots** start their logistics tasks by **loading the necessary medical supplies** from the warehouse into carriers, factoring in the time required for loading operations. The **movement between locations is optimized** by considering travel times based on the speed of each robot.

Once the deliveries begin, the planner efficiently **schedules actions in parallel**, allowing multiple robots to operate simultaneously when feasible. **Medical supplies are unloaded at the correct units**,

and the robots dynamically adjust their routes to optimize the sequence of deliveries. The computed plan carefully **minimizes idle time**, ensuring that at no moment a robot is waiting unnecessarily. The entire logistics workflow is executed with **high efficiency**, leveraging the **durative action constraints** to optimize scheduling.

The search statistics indicate a **more complex problem** than the previous ones. The final plan execution effectively **balances movement, loading, and delivery actions** while ensuring that all goals are met within the shortest feasible time. Future work could explore additional refinements in **parallel execution** and better balancing of **time-dependent constraints** to further reduce overall execution time.

4.5. Problem 5

The implementation of Problem 5 in **PlanSys2** introduced several challenges due to the constraints of the available planning framework. While Problem 4 relied on the **OPTIC planner** to support advanced features such as numeric fluents and dynamic duration calculations, the transition to **POPF 1.1** required significant adjustments to the problem formulation.

A key issue encountered was related to the **POPF 1.1 planner** within PlanSys2, which lacks support for numeric fluents and complex hierarchical structures. These limitations necessitated an alternative representation of problem constraints, particularly for **resource management and execution timing**. Instead of leveraging numerical functions, capacities and inventory levels were tracked using a set of discrete predicates.

Beyond structural changes, inconsistencies between different versions of **POPF** led to unexpected runtime errors. While the problem formulation functioned correctly under **POPF 2.0** (as tested in the planutils Docker environment), the same model caused **segmentation faults in PlanSys2's ROS Docker implementation**. The issue was traced back to the `take_patient` action, where a missing precondition check for the `patient_at_location` predicate caused crashes in **POPF 1.1**. Since **POPF 2.0** has a more robust precondition handling mechanism, this inconsistency was not initially detected.

Another major difficulty was the **non-convergence of plans** in **POPF 1.1**. Even with the simplified domain, the planner struggled to generate complete solutions when handling larger logistics tasks. This was particularly evident when attempting to replicate Problem 4's full-scale scenario, where the number of supply boxes exceeded the threshold that **POPF 1.1** could effectively process. As a result, **some goals had to be removed** in order to achieve feasible execution within PlanSys2.

Despite these challenges, adapting the problem to **PlanSys2** while maintaining logical consistency with Problem 4 demonstrated the trade-offs required when working with different planning frameworks. The necessary modifications highlight the constraints of using **POPF 1.1** within PlanSys2 and underscore the importance of selecting a planner that aligns with the problem's complexity.

Figure 6: Snippet of code execution

5. Conclusions

This report presented the implementation of an automated planning solution for healthcare logistics, utilizing PDDL/HDDL and the ROS2-based PlanSys2 framework. The study systematically explored a series of progressively complex planning problems, incorporating hierarchical task networks (HTN), durative actions, and multi-agent coordination to optimize the delivery of medical supplies and patient transportation in a structured healthcare environment.

The experimental results confirm that automated planning effectively models and addresses logistics challenges in healthcare facilities, ensuring efficient resource utilization while respecting spatial and capacity constraints. The integration of HTN facilitated structured task decomposition, improving modularity and scalability, while durative actions enabled optimized task execution through parallel scheduling.

Despite these successes, the study also revealed certain limitations, particularly related to the restricted capabilities of the POPF 1.1 planner within PlanSys2. These constraints prevented the full implementation of advanced temporal planning features, necessitating modifications for compatibility. This highlights the impact of planner selection on implementation feasibility and solution quality.

Overall, this work demonstrates the feasibility and advantages of AI-driven automated planning in healthcare logistics, providing a structured and effective approach for managing supply distribution and patient

transportation in a controlled environment.

6. LINK

https://github.com/ChiaraMuss/Automatic-Planning